Régi anyag:
https://docs.google.com/presentation/d/1_nmA1F4ag_O-qlJAE4TfVuyfLgSruZLW/edit?usp=sharing&ouid=110539736176923279178&rtpof=true&sd=true

Adatbázisok története és fejlődése

Relációs adatbázisoktól a NoSQL-en át a vektor adatbázisokig

1. Bevezetés

Az adatbázisok az informatikai rendszerek alapvető elemei, amelyek lehetővé teszik az adatok hatékony tárolását, visszakeresését, rendezését és módosítását. A modern világban hatalmas mennyiségű adat keletkezik másodpercenként, és ezek strukturált tárolása elengedhetetlen a döntéstámogatáshoz, automatizáláshoz és gépi tanuláshoz.

Miért van szükség adatbázisokra?

- Az adatok szervezett tárolása lehetővé teszi azok gyors visszakeresését és elemzését.
- Több felhasználó egyidejűleg dolgozhat ugyanazon adatokkal (konkurens hozzáférés).
- Biztosítani lehet az **adatintegritást** és a **hozzáférés-ellenőrzést**.
- Az adatok rendszeres **mentése és visszaállítása** egyszerűbbé válik.
- Automatizált folyamatokhoz (pl. webalkalmazások, gépi tanulás) szükséges egy megbízható háttértároló.

Példa: Egy webshopban több ezer termék, rendelés és felhasználói adat van. Ezek kereshető tárolása adatbázis nélkül gyakorlatilag lehetetlen lenne.

Adat vs. információ

- **Adat**: nyers tények, mért vagy rögzített értékek, amelyek még nem feltétlenül bírnak jelentéssel.
  - Példa: `42`, `2025-05-07`, `„Piros”`
- **Információ**: értelmezett adat, amely kontextusba helyezve új tudást jelent.
  - Példa: „42 éves a felhasználó”, vagy „A rendelés dátuma: 2025-05-07”

Az adat tehát az alap, amiből – megfelelő feldolgozással – információt nyerhetünk. Az adatbázis célja az adatok strukturált tárolása annak érdekében, hogy információvá alakíthassuk őket.

Strukturált, félstrukturált és strukturálatlan adatok

- **Strukturált adat**:
  - Táblázatos formában rendezett, előre meghatározott sémával rendelkezik.
  - Példa: SQL adatbázisok, Excel-táblák, CRM rendszerek.
  - Jellemző: gyors kereshetőség, jól lekérdezhető.

- **Félstrukturált adat**:
- Nincs fix sémája, de tartalmaz címkéket vagy metaadatokat.
- Példa: XML, JSON, YAML fájlok, logok.
- Rugalmasabb, de nehezebb indexelni és lekérdezni.

- **Strukturálatlan adat**:
  - Nincs formázása, nem egyértelmű a tartalma.
  - Példa: szövegfájlok, képek, videók, e-mailek, hangfelvételek.
  - Ezeket általában keresőmotorokkal vagy mesterséges intelligenciával lehet feldolgozni (pl. szöveg- vagy képelemzés).

Az adatbázisok fejlődése szorosan összefügg azzal, hogyan és milyen típusú adatokat akarunk hatékonyan kezelni.

2. Relációs adatbázisok (RDBMS)

- **Történet**:
  - Edgar F. Codd (1970) – relációs modell
  - SQL szabványosítása (1970-es, 1980-as évek)
- **Jellemzők**:
  - Táblák (sorok és oszlopok)
  - Szigorú séma (schema-on-write)
  - ACID tranzakciók: Atomicity, Consistency, Isolation, Durability
- **Példák**: MySQL, PostgreSQL, Oracle, Microsoft SQL Server
- **Előnyök**: Adatintegritás, stabilitás
- **Hátrányok**: Skálázási nehézségek, strukturálatlan adatok kezelése nehézkes

3. NoSQL adatbázisok

- **Megjelenés oka**:
  - Big Data, Web 2.0, gyorsan változó struktúrák
  - Horizontális skálázás igénye
- **Típusai**:
  - Dokumentum-orientált (pl. MongoDB)
  - Kulcs-érték tárolók (pl. Redis, DynamoDB)
  - Oszlop-orientált (pl. Cassandra)
  - Gráf adatbázisok (pl. Neo4j)
- **Jellemzők**:
  - Séma nélküli (schema-on-read)
  - BASE modell: Basically Available, Soft state, Eventually consistent
  - Nagy teljesítmény, skálázhatóság

4. NewSQL adatbázisok

- **Cél**: Relációs modell + NoSQL skálázhatóság
- **Jellemzők**:

    - ACID + elosztott tranzakciók
- **Példák**:
    - Google Spanner
    - CockroachDB
    - VoltDB

5. Vektor adatbázisok

- **Motiváció**:
    - Gépi tanulás, jelentés szerinti keresés
    - Embeddingek (szövegek, képek, hangok reprezentációja)
- **Működés**:
    - Objektum → Vektor (embedding)
    - Közelségi keresés: k-nn (legközelebbi szomszédok)
- **Példák**:
    - FAISS (Facebook)
    - Milvus
    - Weaviate
    - Pinecone
    - Chroma (LLM-es RAG rendszerekhez)
- **Alkalmazások**:
    - Dokumentumkeresés (RAG)
    - Képkeresés
    - Ajánlórendszerek

6. Tendenciák és jövőkép

- Multimodális adatbázisok (szöveg + kép + struktúrált adatok)
- Hybrid keresés: SQL + vektoros keresés együtt
- AI-native rendszerek: LLM + adatbázis integráció (pl. LangChain)

7. Összefoglaló táblázat

Típus	Példa	Előnyök	Hátrányok
Relációs	PostgreSQL	Stabil, jól ismert	Nehezen skálázható
NoSQL	MongoDB	Rugalmas, horizontálisan skálázható	Nincs szigorú séma, konzisztencia lehet gyenge
NewSQL	CockroachDB	ACID + skálázhatóság	Új technológia, kisebb ökoszisztéma
Vektor	FAISS	Jelentésalapú keresés, ML támogatás	Nem SQL-alapú lekérdezések, új gondolkodásmód

8. Demó

- SQL lekérdezés vs. MongoDB lekérdezés
- Szöveg → embedding → hasonló dokumentumok keresése vektor alapján